Flow Visualization

Part 2 (of 3)

Retrospect: Flow Visualization, Part 1
- introduction, overview
  - simulation vs. measurement vs. modelling
  - 2D vs. surfaces vs. 3D
  - steady vs time-dependent
  - direct vs. indirect FlowVis
- experimental FlowVis
  - general possibilities
  - PIV + example
- visualization of models
- flow visualization with arrows

Overview: Flow Visualization, Part 2
- numerical integration
  - Euler-integration
  - Runge-Kutta-integration
- streamlines
  - in 2D
  - particle paths
  - in 3D, sweeps
  - illuminated streamlines
- streamline placement
Integration of Streamlines

Numerical Integration

Streamlines – Theory

- Correlations:
  - flow data v: derivative information
  - $\frac{dx}{dt} = v(x)$: spatial points $x \in \mathbb{R}^n$, time $t \in \mathbb{R}$, flow vectors $v \in \mathbb{R}^n$
  - streamline $s$: integration over time, also called trajectory, solution, curve
  - $s(t) = s_0 + \int_{0}^{t} v(s(u)) \, du$; seed point $s_0$, integration variable $u$
  - difficulty: result $s$ also in the integral $\Rightarrow$ analytical solution usually impossible!

Streamlines – Practice

- Basic approach:
  - theory: $s(t) = s_0 + \int_{0}^{t} v(s(u)) \, du$
  - practice: numerical integration
  - idea: (very) locally, the solution is (approx.) linear
  - Euler integration:
    - follow the current flow vector $v(s)$ from the current streamline point $s$, for a very small time ($dt$) and therefore distance
    - Euler integration: $s_{i+1} = s_i + dt \cdot v(s_i)$, integration of small steps ($dt$ very small)
Euler Integration – Example

2D model data:

\[ v_x = \frac{dx}{dt} = -y \]
\[ v_y = \frac{dy}{dt} = \frac{x}{2} \]

Sample arrows:

True solution: ellipses!

Euler Integration – Example

Seed point \( s_0 = (0\cdot-1) \); current flow vector \( v(s_0) = (1\cdot0) \); \( \Delta t = 1/2 \)

Euler Integration – Example

New point \( s_1 = s_0 + v(s_0) \cdot \Delta t = (1/2\cdot1) \); current flow vector \( v(s_1) = (1\cdot1/4) \);
Euler Integration – Example

- New point \( s_2 = s_1 + v(s_1) \cdot dt = (1|-7/8)^T \);
  current flow vector \( v(s_2) = (7/8|1/2)^T \);

---

Euler Integration – Example

- \( s_3 = (23/16|-5/8)^T \approx (1.44|-0.63)^T \);
  \( v(s_3) = (5/8|23/32)^T \approx (0.63|0.72)^T \);

---

Euler Integration – Example

- \( s_4 = (7/4|-17/64)^T \approx (1.75|-0.27)^T \);
  \( v(s_4) = (17/64|7/8)^T \approx (0.27|0.88)^T \);
Euler Integration – Example

\[ s_9 \approx (0.20|1.69)^T; \]
\[ v(s_9) \approx (-1.69|0.10)^T; \]

clearly: large integration error, \( dt \) too large!
19 steps
Euler Integration – Example

- dt smaller (1/4): more steps, more exact!
  \( s_{36} \approx (0.04|\ -1.74)^T; v(s_{36}) \approx (1.74|\ 0.02)^T; \)
- 36 steps

Comparison Euler, Step Sizes

Euler is getting better proportionally to \( dt \)

Euler Example – Error Table

<table>
<thead>
<tr>
<th>( dt )</th>
<th>#steps</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/2</td>
<td>19</td>
<td>~200%</td>
</tr>
<tr>
<td>1/4</td>
<td>36</td>
<td>~75%</td>
</tr>
<tr>
<td>1/10</td>
<td>89</td>
<td>~25%</td>
</tr>
<tr>
<td>1/100</td>
<td>889</td>
<td>~2%</td>
</tr>
</tbody>
</table>
| 1/1000  | 8889   | ~0.2%   | ✔️
Better than Euler Integr.: RK

Runge-Kutta Approach:
- theory: \( s(t) = s_0 + \int_{0}^{t} v(s(u)) \, du \)
- Euler: \( s_i = s_0 + \sum_{0}^{i} v(s_u) \cdot \Delta t \)
- Runge-Kutta integration:
  - idea: cut short the curve arc
- RK-2 (second order RK):
  1: do half a Euler step
  2: evaluate flow vector there
  3: use it in the origin
- RK-2 (two evaluations of \( v \) per step):
  \( s_{i+1} = s_i + v(s_i) d\tau / 2 \cdot d\tau \)

RK-2 Integration – One Step

Seed point \( s_0 = (0| -2) \ T; \)
current flow vector \( v(s_0) = (2|0) \ T; \)
preview vector \( v(s_0 + v(s_0) d\tau / 2) = (2|0.5) \ T; \)
\( d\tau = 1 \)

RK-2 – One more step

Seed point \( s_1 = (2| -1.5) \ T; \)
current flow vector \( v(s_1) = (1.5|1) \ T; \)
preview vector \( v(s_1 + v(s_1) d\tau / 2) = (1|1.4) \ T; \)
\( d\tau = 1 \)
RK-2 – A Quick Round

- RK-2: even with $dt=1$ (9 steps) better than Euler with $dt=1/8$ (72 steps)

Euler vs. Runge-Kutta

- RK-4: pays off only with complex flows
- Here approx. like RK-2

RK-4 vs. Euler, RK-2

- Even better: fourth order RK:
  - four vectors $a$, $b$, $c$, $d$
  - one step is a convex combination:
    $$ s_{i+1} = s_i + \frac{(a + 2 \cdot b + 2 \cdot c + d)}{6} $$
  - vectors:
    - $a = dt \cdot v(s_i)$ ... original vector
    - $b = dt \cdot v(s_i + a/2)$ ... RK-2 vector
    - $c = dt \cdot v(s_i + b/2)$ ... use RK-2 ...
    - $d = dt \cdot v(s_i + c)$ ... and again!

Helwig Hauser 22

Helwig Hauser 23

Helwig Hauser 24
Integration, Conclusions

Summary:
- analytic determination of streamlines usually not possible
- hence: numerical integration
- several methods available (Euler, Runge-Kutta, etc.)
- Euler: simple, imprecise, esp. with small dt
- RK: more accurate in higher orders
- furthermore: adaptive methods, implicit methods, etc.

Flow Visualization with Streamlines

Streamlines, Particle Paths, etc.

Streamlines in 2D

Adequate for overview
**Visualization with Particles**

- Particle paths = streamlines (steady flows)
- Variants (time-dependent data):
  - streak lines: steadily new particles
  - path lines: long-term path of one particle

**Streamlines in 3D**

- Color coding: Speed
- Selective Placement

**3D Streamlines with Sweeps**

- Sweeps: better spatial 3D perception
Streamline Placement

in 2D

Problem: Choice of Seed Points

- Streamline placement:
  - If regular grid used: very irregular result
Overview of Algorithm
- Idea: streamlines should not get too near to each other
- Approach:
  - choose a seed point with distance \( d_{\text{sep}} \) from an already existing streamline
  - forward- and backward-integration until distance \( d_{\text{test}} \) is reached (or ...).
  - two parameters:
    - \( d_{\text{sep}} \) ... start distance
    - \( d_{\text{test}} \) ... minimum distance

Algorithm – Pseudocode
- Compute initial streamline, put it into a queue
- Initial streamline becomes current streamline
- WHILE not finished DO:
  - TRY: get new seed point which is \( d_{\text{sep}} \) away from current streamline
  - IF successful THEN compute new streamline and put to queue
  - ELSE IF no more streamline in queue THEN exit loop
  - ELSE next streamline in queue becomes current streamline

Streamline Termination
- When to stop streamline integration:
  - when dist. to neighboring streamline \( \leq d_{\text{test}} \)
  - when streamline leaves flow domain
  - when streamline runs into fixed point (v=0)
  - when streamline gets too near to itself
  - after a certain amount of maximal steps
New Streamlines

Different Streamline Densities

- Variations of $d_{\text{sep}}$ in rel. to image width:
  - 6%  
  - 3%  
  - 1.5%

$d_{\text{sep}}$ vs. $d_{\text{test}}$

- $d_{\text{test}} = 0.9 \cdot d_{\text{sep}}$
- $d_{\text{test}} = 0.5 \cdot d_{\text{sep}}$
Tapering and Glyphs

- Thickness in rel. to dist.

\[
\frac{d - d_{sep}}{d_{sep}}: \forall d \geq d_{sep}\]
\[
\frac{d_{sep} - d_{sep}}{d_{sep}}: \forall d < d_{sep}
\]

- Directional glyphs:

Literature

- Paper (more details):

Acknowledgements

- For material used in this lecture:
  - Bruno Jobard
  - Malte Zöckler
  - Georg Fischel
  - Frits Post
  - Roger Crawfis
  - myself... ;-) (i.e., Helwig Hauser)
  - etc.